# نظریه بازیها
# Game Theory

**ارائه کننده: امیرحسین نیکوفرد**
**مهندسی برق و کامپیوتر دانشگاه خواجه نصیر**

دانشگاه صنعتی خواجه نصیرالدین طوسی

# InfiniteDynamic Games

Material

- Dynamic Non-cooperative Game Theory: Second Edition
  - Chapter5: Sections 5:5 and Chapter6: Sections 6:2
- An Introductory Course in Non-cooperative Game Theory
  - Chapter 16,17

# InfiniteDynamic Games

- Zero sum games
- Non-zero sum games
- Infinite Games
- **Infinite Dynamic Games**
  - Dynamic games in discrete time
  - Information structures
  - Continuous-time differential games
  - Discrete-time dynamic programming
  - **Continuous-time dynamic programming**
  - **Discrete-time dynamic programming for zero sum games**

# Differential games with variable termination time

Consider now a one-player continuous-time differential game with the usual dynamics

$$\underbrace{\dot{x}(t)}_{\substack{\text{state}\\ \text{derivative}}} = \underbrace{f}_{\substack{\text{game}\\ \text{dynamics}}}(\underbrace{t}_{\text{time}}, \underbrace{x(t)}_{\substack{\text{current}\\ \text{state}}}, \underbrace{u(t)}_{\substack{P_1\text{'s action}\\ \text{at time } t}}), \qquad x(t) \in R^n, u(t) \in U, t \geq 0 \qquad (1)$$

and initialized at a given $x(0) = x_0$, but with an integral cost with variable horizon:

$$J = \int_0^{T_{end}} \underbrace{g(t, x(t), u(t))dt}_{\text{cost along trajctory}} + \underbrace{q(T_{end}, x(T_{end}))}_{\text{final cost}} \qquad (2)$$

- where $T_{end}$ is the first time at which the state $x(t)$ enters a closed set $\chi_{end} \subset R^n$ or $T_{end} = \infty$ in case $x(t)$ never enters $\chi_{end}$.

# Differential games with variable termination time

- For this game, the *cost-to-go from state x at time* $\tau$ is defined by

$$V(\tau,x) := \inf_{u(t)\in U,\forall t\geq\tau} \int_{\tau}^{T_{end}} g(t,x(t),u(t))dt + q(T_{end},x(T_{end})) \qquad (3)$$

where the state $x(t)$, $t \geq \tau$ satisfies the dynamics

$$x(\tau) = \mathrm{x} \qquad\qquad \dot{x} = f(\mathrm{t},\mathrm{x}(\mathrm{t}),\mathrm{u}(\mathrm{t})), \qquad \forall t \geq \tau$$

and $T_{end}$ denotes the first time at which $x(t)$ enters the closed set $\chi_{end}$ When we compute $V(\tau,\mathrm{x})$ for some $x \in \chi_{end}$, we have $T_{end} = \tau$ and therefore

$$V(\tau,\mathrm{x}) = \mathrm{q}(\tau,\mathrm{x}), \qquad \forall\tau > 0, x \in \chi_{end} \qquad (4)$$

# Differential games with variable termination time

instead of the boundary condition in Continuous-time dynamic programming with fixed termination time. However, it turns out that the Hamilton-Jacobi-Bellman equation is still the same and we have the following result:

**Theorem :** Any continuously differentiable function $V(\tau,x)$ that satisfies the Hamilton-Jacobi-Bellman equation with

$$V(\tau,\mathrm{x}) = \mathrm{q}(\tau,\mathrm{x}), \qquad \forall \tau > 0, x \in \chi_{end}$$

is equal to the cost-to-go $V(\tau,x)$. In addition, if the infimum in the Hamilton-Jacobi-Bellman equation is always achieved at some point in $U$, we have that:

# Differential games with variable termination time

1. For any given $x_0$, an optimal open-loop policy $\gamma^{OL}$ is given by

$$\gamma^{OL}(t,x_0) := u^*(t), \qquad \forall \mathrm{t} \in [0,\mathrm{T}]$$

with $u^*(t)$ obtained from solving

$$u^*(t) = \arg\min_{u \in U} g(\mathrm{t},x^*(\mathrm{t}),u) + \frac{\partial V(t,x^*(\mathrm{t}))}{\partial x} f(t,x^*(\mathrm{t}),u)$$

$$\dot{x}^*(\mathrm{t}) = f(\tau,x^*(\mathrm{t}),u^*(t)), \qquad \forall t \in [0,\mathrm{T}_{end}], \qquad \mathrm{x}^*(0) = x_0$$

2. An optimal (time consistent) state-feedback policy $\gamma^{FB}$ is given by

$$\gamma^{FB}(t,x(t)) = \arg\min_{u \in U} g(\tau,x(\mathrm{t}),u) + \frac{\partial V(\tau,x(\mathrm{t}))}{\partial x} f(\tau,x(\mathrm{t}),u) \ \forall \mathrm{t} \in [0,\mathrm{T}_{end}]$$

Either of the above optimal policies leads to an optimal cost equal to $V(0, x_0)$.

# Differential games with variable termination time

- The Hamilton-Jacobi-Bellman equation is a partial differential equation (PDE) and (4) can be viewed as a boundary condition for this PDE
- When we can find a continuously differentiable solution to this PDE that satisfies the appropriate boundary condition, we automatically obtain the cost-to-go. Unfortunately, solving a PDE is often difficult to solve and many times the HJB equation does not have continuously differentiable solutions
- Open-loop and state-feedback information structures are "optimal," in the sense that it is not possible to achieve a cost lower than $V(0, x_0)$, regardless of the information structure.

# Differential games with variable termination time

**Proof of Theorem:** Let $u^*(t)$ and $x^*(t)$ $\forall t \geq 0$ be a trajectory arising from either the open-loop or the state-feedback policies and let $\bar{u}(t)$ and $\bar{x}(t), \forall t \geq 0$ be another arbitrary trajectory. To prove optimality, we need to show that the latter trajectory cannot lead to a cost lower than the former.

Since $V(\tau, x)$ satisfies the Hamilton-Jacobi-Bellman equation and $u^*(t)$ achieves the infimum in the Hamilton-Jacobi-Bellman equation, for every $\forall t \in [0, T]$, we have that

$$0 = \inf_{u \in U}\left(g(t, x^*(t), u) + \frac{\partial V(\tau, x^*(t))}{\partial \tau} + \frac{\partial V(t, x^*(t))}{\partial x} f(t, x^*(t), u)\right)$$

$$= g(t, x^*(t), u^*(t)) + \frac{\partial V(t, x^*(t))}{\partial t} + \frac{\partial V(t, x^*(t))}{\partial x} f(t, x^*(t), u^*(t)) \quad (5)$$

# Differential games with variable termination time

However, since $\bar{u}(t)$ does not necessarily achieve the infimum, we have that

$$
\begin{aligned}
0 &= \inf_{u\in U}\left(g(t,\bar{x}(t),u) + \frac{\partial V(t,\bar{x}(t))}{\partial t} + \frac{\partial V(t,\bar{x}(t))}{\partial x} f(t,\bar{x}(t),u)\right) \\
&\le g(t,\bar{x}(t),\bar{u}(t)) + \frac{\partial V(t,\bar{x}(t))}{\partial t} + \frac{\partial V(t,\bar{x}(t))}{\partial x} f(t,\bar{x}(t),\bar{u}(t)) \qquad (6)
\end{aligned}
$$

Let $T^*_{end} \in [0,\infty]$ and $\bar{T}_{end} \in [0,\infty]$ denote the times at which $x^*(t)$ and $\bar{x}(t)$ respectively, enter the set $\chi_{end}$. Integrating both side of (5) and (6) over the intervals $[0,T^*_{end}]$ and $[0,\bar{T}_{end}]$, respectively.

# Differential games with variable termination time

we conclude that

$$0=\int_0^{T^*_{end}}\Big(g(t,x^*(t),u^*(t))+\underbrace{\frac{\partial V(t,x^*(t))}{\partial t}+\frac{\partial V(t,x^*(t))}{\partial x}f(t,x^*(t),u^*(t))}_{\frac{dV(\mathrm{t},x^*(\mathrm{t}))}{dt}}\Big)dt$$

$$\le\int_0^{\bar{T}_{end}}\Big(g(t,\bar{x}(t),\bar{u}(t))+\underbrace{\frac{\partial V(t,\bar{x}(t))}{\partial t}+\frac{\partial V(\tau,\bar{x}(t))}{\partial x}f(t,\bar{x}(t),\bar{u}(t))}_{\frac{dV(t,\bar{x}(\mathrm{t}))}{dt}}\Big)\mathrm{dt}$$

# Differential games with variable termination time

from which we obtain

$$0 = \int_0^{T_{end}^*} g(t, x^*(t), u^*(t))\,\mathrm{dt} + V(T_{end}^*, x^*(T_{end}^*)) - V(0, x_0)$$

$$\leq \int_0^{\bar{T}_{end}} g(t, \bar{x}(t), \bar{u}(t))\,\mathrm{dt} + V(\bar{T}_{end}, \bar{x}(\bar{T}_{end})) - V(0, x_0)$$

Using boundary condition , two conclusions can be drawn from here:

First, the signal $\bar{u}(t)$ does not lead to a cost smaller than that of $u^*(t)$, because

$$\int_0^{T_{end}^*} g(t, x^*(t), u^*(t))\,\mathrm{dt} + q(T_{end}^*, x^*(T_{end}^*))$$

$$\leq \int_0^{\bar{T}_{end}} g(t, \bar{x}(t), \bar{u}(t))\,\mathrm{dt} + q(\bar{T}_{end}, \bar{x}(\bar{T}_{end}))$$

# Differential games with variable termination time

Second, $V(0,x_0)$ is equal to the optimal cost obtained with $u^*(t)$, because

$$V(0,x_0) = \int_0^{T_{end}^*} g(t,x^*(t),u^*(t))\,\mathrm{dt} + q(T_{end}^*, x^*(T_{end}^*))$$

If we had carry out the above proof on an interval $[\tau, \mathrm{T}]$ with initial state $x(\tau) = \mathrm{x}$, we would have concluded that $V(\tau, x)$ is the (optimal) value of the cost-to-go from state $x$ at time $\tau$.

- In a open-loop setting both $u^*(t)$ and $x^*(t), \forall\, \mathrm{t} \in [0, T_{end}]$ are pre-computed before the game starts.
- Both the open-loop and the state-feedback policies lead precisely to the same trajectory.

# InfiniteDynamic Games

- Zero sum games
- Non-zero sum games
- Infinite Games
- **Infinite Dynamic Games**
  - Dynamic games in discrete time
  - Information structures
  - Continuous-time differential games
  - Discrete-time dynamic programming
  - Continuous-time dynamic programming
  - **Discrete-time dynamic programming for zero sum games**

# Dynamic games in discrete time

## Zero-sum dynamic games in discrete time:

We now discuss solution methods for two-player zero-sum dynamic games in discrete time, which corresponds to dynamics of the form

$$\underbrace{x_{k+1}}_{\substack{entry\ node \\ at\ stage\ k+1}} = \underbrace{f_k}_{\substack{\text{"dynamics" at} \\ stage\ k}} \Big( \underbrace{x_k}_{\substack{entry\ node \\ at\ stage\ k}} , \underbrace{u_k}_{\substack{P_1\text{'s action} \\ at\ stage\ k}} , \underbrace{d_k}_{\substack{P_2\text{'s action at} \\ stage\ k}} \Big) \qquad , \forall \mathrm{k} \in \{1, 2, \dots, \mathrm{K}\}$$

starting at some initial state $x_1$ in the state space $\chi$. At each time $k$, $\mathrm{P}_1$'s action $u_k$ is required to belong to an action space $U_k$ and $\mathrm{P}_2$'s action $d_k$ is required to belong to an action space $D_k$. We assume a finite horizon $(K < \infty)$ stage additive costs of the form

$$J = \sum_{k=1}^{K} g(\mathrm{x}_k, \mathrm{u}_k, \mathrm{d}_k) \qquad (7)$$

# Dynamic games in discrete time

that $P_1$ wants to minimize and $P_2$ wants to maximize. In this part we consider a **state-feedback information structure**, which correspond to policies of the form

$$u_k = \gamma_k(x_k), \qquad ,d_k = \sigma_k(x_k), \qquad \forall \mathrm{k} \in \{1,2,\ldots,\mathrm{K}\},$$

Suppose that for a given state-feedback policy $\gamma$ for $P_1$ and a given state-feedback policy $\sigma$ for $P_2$, we denote by $J(\gamma,\sigma)$ ,the corresponding value of the cost (7). Our goal is to find a saddle-point pair of equilibrium policies $(\gamma^*,\sigma^*)$ for which

$$J(\gamma^*,\sigma) \le J(\gamma^*,\sigma^*) \le J(\gamma,\sigma^*), \qquad \forall \gamma \in \Gamma_1, \forall \sigma \in \Gamma_2,$$

# Dynamic games in discrete time

where $\Gamma_1$ and $\Gamma_2$ denote the sets of all state-feedback policies for $P_1$ and $P_2$, respectively. Re-writing previous equation as

$$J(\gamma^*,\sigma^*) = \min_{\gamma\in\Gamma_1} J(\gamma,\sigma^*), \qquad J(\gamma^*,\sigma^*) = \max_{\sigma\in\Gamma_2} J(\gamma^*,\sigma)$$

we conclude that if $\sigma^*$ was known we could obtain $\gamma^*$ from the following single-player optimization

$$\textit{minimize over } \gamma \in \Gamma_1 \textit{ the cost} \quad J(\gamma,\sigma^*) := \sum_{k=1}^{K} g(\mathrm{x}_k, \mathrm{u}_k, \sigma^*_k(\mathrm{x}_k))$$

$$\textit{subject to the dynamics} \quad x_{k+1} = f_k(x_k, u_k, \sigma^*_k(\mathrm{x}_k))$$

# Dynamic games in discrete time

In view of what we saw in Lecture **14**, an optimal state-feedback policy $\gamma^*$ could be constructed first using a backward iteration to compute the cost-to-go $V_k^1(\mathrm{x})$ for $\mathrm{P}_1$ using

$$V_{K+1}^1(\mathrm{x}) = 0, \quad V_k^1(\mathrm{x}) = \inf_{u_k \in U_k} \left(g_k(\mathrm{x}, \mathrm{u}_k, \sigma^*_k(\mathrm{x})) + V_{k+1}^1(f_k(x, u_k, \sigma^*_k(x)))\right)$$

$$, \quad \forall k \in \{1, 2, \ldots, \mathrm{K}\} \qquad (8)$$

and then

$$\gamma_k^*(\mathrm{x}) = \arg\min_{u_k \in U_k} (g_k(\mathrm{x}, \mathrm{u}_k, \sigma^*_k(\mathrm{x})) + V_{k+1}(f_k(x, u_k, \sigma^*_k(x))))$$

$$, \quad \forall k \in \{1, 2, \ldots, \mathrm{K}\} \qquad (9)$$

Moreover, the minimum $\mathrm{J}(\gamma^*, \sigma^*)$ is given by $V_1^1(\mathrm{x}_1)$

# Dynamic games in discrete time

Similarly, if $\gamma^*$ was known we could obtain an optimal state-feedback policy $\sigma^*$ from the following single-player optimization

$$\textit{maximize over } \forall\sigma\in\Gamma_2 \textit{ the cost} \qquad J(\gamma^*,\sigma) := \sum_{k=1}^{K} g(\mathrm{x}_k,\gamma_k^*(\mathrm{x}_k),d_k)$$

$$\textit{subject to the dynamics} \qquad x_{k+1} = f_k(x_k,\gamma_k^*(\mathrm{x}_k),d_k)$$

**Note:** We only derived the dynamic programming equations for single-player minimizations, but for single-player maximizations analogous formulas are still valid, provided that we replace all infima by suprema

# Dynamic games in discrete time

In view of what we saw in Lecture **14**, an optimal state-feedback policy $\sigma^*$ could be constructed first using a backward iteration to compute the cost-to-go $V_k^2(\mathrm{x})$ for $\mathrm{P}_2$ using

$$V_{K+1}^2(\mathrm{x}) = 0, \quad V_k^2(\mathrm{x}) = \sup_{d_k \in D_k} \left( g_k(\mathrm{x}, \gamma_k^*(\mathrm{x}), d_k) + V_{k+1}^2(f_k(x, \gamma_k^*(\mathrm{x}), d_k(x))) \right), \quad \forall k \in \{1, 2, \ldots, \mathrm{K}\} \qquad (10)$$

and then

$$\sigma^*{}_k(\mathrm{x}) = \arg\max_{d_k \in D_k} \left( g_k(\mathrm{x}, \gamma_k^*(\mathrm{x}), d_k) + V_{k+1}^2(f_k(x, \gamma_k^*(\mathrm{x}), d_k(x))) \right), \quad \forall k \in \{1, 2, \ldots, \mathrm{K}\} \qquad (11)$$

Moreover, the minimum $\mathrm{J}(\gamma^*, \sigma^*)$ is given by $V_2^1(\mathrm{x}_1)$

# Discrete-time dynamic programming

The key to finding the saddle-point pair of equilibrium policies $(\gamma^*, \sigma^*)$ is to realize that it is possible to construct a pair of state-feedback policies for which the four equations (8), (9), (10), (11) all hold.

To see how this can be clone, consider the costs-to-go $V_K^1(\mathrm{x})$, $V_K^2(\mathrm{x})$ and state-feedback policies $\gamma_K^*, \sigma_K^*$ at the last stage. For (8), (9), (10), and (11) to hold we need that

$$V_K^1(\mathrm{x}) = \inf_{u_K \in U_K} (g_K(\mathrm{x}, \mathrm{u}_K, \sigma^*{}_K(\mathrm{x}))) \qquad \gamma_K^*(\mathrm{x}) = \arg\min_{u_K \in U_K}(g_K(\mathrm{x}, \mathrm{u}_K, \sigma^*{}_K(\mathrm{x})))$$

$$V_K^2(\mathrm{x}) = \sup_{d_K \in D_K} (g_K(\mathrm{x}, \gamma_K^*(\mathrm{x}), d_K)) \qquad \sigma^*{}_K(\mathrm{x}) = \arg\max_{d_K \in D_K}(g_K(\mathrm{x}, \gamma_K^*(\mathrm{x}), d_K))$$

# Discrete-time dynamic programming

which can be re-written equivalently

$$V_K^1(\mathrm{x}) = g_K(\mathrm{x}, \gamma_K^*(\mathrm{x}), \sigma^*{}_K(\mathrm{x})) \le g_K(\mathrm{x}, \mathrm{u}_K, \sigma^*{}_K(\mathrm{x})) \qquad \forall u_K \in U_K$$

$$V_K^1(\mathrm{x}) = g_K(\mathrm{x}, \gamma_K^*(\mathrm{x}), \sigma^*{}_K(\mathrm{x})) \ge g_K(\mathrm{x}, \gamma_K^*(\mathrm{x}), d_K) \qquad \forall d_K \in D_K$$

Since the right-hand side of the top and bottom equalities are the same, we conclude that $V_K^1(\mathrm{x}) = V_K^1(\mathrm{x})$. Moreover, this shows that the pair $(\gamma_K^*(\mathrm{x}), \sigma^*{}_K(\mathrm{x})) \in U_K \times D_K$ must be a saddle-point equilibrium for the zero-sum game with outcome

$$g_K(\mathrm{x}, \mathrm{u}_K, d_K)$$

# Discrete-time dynamic programming

and actions $u_K \in U_K$ and $d_K \in D_K$ for the minimizer and maximizer, respectively. Moreover, $V_K^1(\mathrm{x}) = V_K^2(\mathrm{x})$ must be precisely equal to the value of this game. In view of the results that we saw for zero-sum games, this will only be possible if security policies exist and the security levels for both players are equal to the value of the game, i.e.,

$$V_K^1(\mathrm{x}) = V_K^2(\mathrm{x}) = V_K(\mathrm{x})$$

$$V_K(\mathrm{x}) = \min_{u_K \in U_K} \sup_{d_K \in D_K} g_K(\mathrm{x}, \mathrm{u}_K, d_K) = \max_{d_K \in D_K} \inf_{u_K \in U_K} g_K(\mathrm{x}, \mathrm{u}_K, d_K)$$

Consider now the costs-to-go $V_{K-1}^1, V_{K-1}^2$ and state-feedback policies $\gamma_{K-1}^*, \sigma^*_{K-1}$ at stage $K$ - 1. For (8), (9), (10), and (11) to hold we need that

# Discrete-time dynamic programming

$$V^1_{K-1}(\mathrm{x}) = \inf_{u_{K-1}\in U_{K-1}} (g_{K-1}(\mathrm{x},\mathrm{u}_{K-1},\sigma^*{}_{K-1}(\mathrm{x})) + V_K(f_{K-1}(x,u_{K-1},\sigma^*{}_{K-1}(x))))$$

$$\gamma^*_{K-1}(\mathrm{x}) = \arg\min_{u_{K-1}\in U_{K-1}} (g_{K-1}(\mathrm{x},\mathrm{u}_{K-1},\sigma^*{}_{K-1}(\mathrm{x})) + V_K(f_{K-1}(x,u_{K-1},\sigma^*{}_{K-1}(x))))$$

$$V^2_{K-1}(\mathrm{x}) = \sup_{d_{K-1}\in D_{K-1}} (g_{K-1}(\mathrm{x},\gamma^*_{K-1}(\mathrm{x}),d_{K-1}) + V_K(f_{K-1}(x,\gamma^*_{K-1}(\mathrm{x}),d_{K-1}(x))))$$

$$\sigma^*{}_{K-1}(\mathrm{x}) = \arg\max_{d_{K-1}\in D_{K-1}} (g_k(\mathrm{x},\gamma^*_{K-1}(\mathrm{x}),d_{K-1}) + V_K(f_k(x,\gamma^*_{K-1}(\mathrm{x}),d_{K-1}(x))))$$

and so we now conclude that the pair $(\gamma^*_{K-1}(\mathrm{x}),\sigma^*{}_{K-1}(x)) \in U_{K-1}\times D_{K-1}$ must be a saddle-point equilibrium for the zero-sum game with outcome

$$g_{K-1}(\mathrm{x},u_{K-1},d_{K-1}) + V_K(f_{K-1}(x,u_{K-1},d_{K-1}(x)))$$

# Discrete-time dynamic programming

and actions $u_{K-1} \in U_{K-1}$ and $d_{K-1} \in D_{K-1}$ for the minimizer and maximizer, respectively. Moreover, $V^1_{K-1}(\mathrm{x}) = V^2_{K-1}(\mathrm{x})$ must be precisely equal to the value of this game. Continuing this reasoning backwards in time all the way to the first stage, we obtain the following result:

**Theorem:** Assume that we can recursively compute functions $, V_1(\mathrm{x}), V_2(\mathrm{x}), \ldots, V_K(\mathrm{x})$, such that $\mathrm{x} \in \chi, \forall \mathrm{k} \in \{1,2,\ldots,\mathrm{K}\}$ we have that

$$
\begin{aligned}
V_k(\mathrm{x}) &:= \min_{u_k \in U_k} \sup_{d_k \in D_k} \left(g_k(\mathrm{x}, u_k, d_k) + V_{k+1}(f_k(x, u_k, d_k(x)))\right) \\
&= \max_{d_k \in D_k} \inf_{u_k \in U_k} \left(g_k(\mathrm{x}, u_k, d_k) + V_{k+1}(f_k(x, u_k, d_k(x)))\right) \qquad (12)
\end{aligned}
$$

# Discrete-time dynamic programming

where

$$V_{K+1}(\mathrm{x}) = 0 \qquad , \mathrm{x} \in \chi$$

Then the pair of policies $(\gamma^*, \sigma^*)$ defined as follows is a saddle-point equilibrium in state-feedback policies:

$$\gamma_k^*(\mathrm{x}) = \arg\min_{u_k \in U_k}(g_k(\mathrm{x}, u_k, d_k) + V_{k+1}(f_k(x, u_k, d_k(x)))) \qquad (13)$$

$$\sigma^*{}_k(\mathrm{x}) = \arg\max_{d_k \in D_k}(g_k(\mathrm{x}, u_k, d_k) + V_{k+1}(f_k(x, u_k, d_k(x)))) \qquad (14)$$

$\mathrm{x} \in \chi, \forall \mathrm{k} \in \{1, 2, \ldots, \mathrm{K}\}$. Moreover, the value of the game is equal to $V_1(x_1)$.

# Discrete-time dynamic programming

- we actually conclude that

1) $P_2$ cannot get a reward larger than $V_1(x_1)$ against $\gamma_k^*(x)$, regardless of the information structure available to $P_2$, and

2) $P_1$ cannot get a cost smaller than $V_1(x_1)$ against $\sigma^*_k(x)$, regardless of the information structure available to $P_1$.

- In practice, this means that $\gamma_k^*(x)$ and $\sigma^*_k(x)$ are "extremely safe" policies for $P_1$ and $P_2$, respectively, since they guarantee a level of reward regardless of the information structure for the other player.

# Solving finite zero-sum games with MATLAB

For games with finite state spaces and finite actions spaces, the backwards iteration in (12) can be implemented very efficiently in MATLAB. To this effect, suppose that we enumerate all states so that the state-space can be viewed as

$$\chi := \{1, 2, \ldots, n_\chi\}$$

and that we enumerate all actions so that the action spaces can be viewed as

$$U := \{1, 2, \ldots, n_U\} \qquad \mathrm{D} := \{1, 2, \ldots, n_D\}$$

For simplicity, we shall assume that all states can occur at every stage and that all actions are also available at every stage.

# Solving finite zero-sum games with MATLAB

In this case, each function $f_k(x, u, d)$ and $g_k(x, u, d)$ that define the game dynamics and the stage-cost, respectively, can be represented by a three-dimensional $n_\chi \times n_U \times n_D$ tensor. On the other hand, each $V_k(x)$ can be represented by a $n_\chi \times 1$ columns vector with one row per state. Suppose then that the following variables are available within MATLAB.

- F is a cell-array with *K* elements, each equal to an $n_\chi \times n_U \times n_D$ three-dimensional matrix so that F{k} represents the game dynamics function $f_k(x, u, d), \forall \mathrm{x} \in \chi, \mathrm{u} \in \mathrm{U}, \mathrm{d} \in \mathrm{D}, \mathrm{k} \in \{1, 2, \ldots, \mathrm{K}\}$.
- G is a cell-array with *K* elements, each equal to an $n_\chi \times n_U \times n_D$ three-dimensional matrix so that G{k} represents the stage-cost function $g_k(x, u, d), \forall \mathrm{x} \in \chi, \mathrm{u} \in \mathrm{U}, \mathrm{d} \in \mathrm{D}, \mathrm{k} \in \{1, 2, \ldots, \mathrm{K}\}$.

# Solving finite zero-sum games with MATLAB

With these definitions, we can construct $V_k(x)$ in (12) very efficiently using the following MATLAB code

```
V{K+1}=zeros(size(G{K},1),1);
for k=K:-1:1
Vminmax=min (max (G{k}+V{k+1} (F{k}) , [] , 3) , [] , 2) ;
Vmaxmin=max (min (G{k }+V{k+1} (F{k}), [] , 2) , [] , 3) ;
 if any(Vminmax-=Vmaxmin)
       error('Saddle-point cannot be found')
     end
V{k}=Vminmax;
end
```

# Solving finite zero-sum games with MATLAB

After running this code, the following variable as been created:

- V is a cell-array with K + 1 elements, each equal to an $n_{\chi} \times 1$ columns vector so that V{k} represents $V_k(x), \forall x \in \chi, k \in \{1, 2, \ldots, K\}$.

For a given state x at stage k, the optimal actions u and d given by (13)-(14) can be obtained using

```
[dummy, u] =min (max (G (x , :  , : )+V{k+1}(F (x , : , : )),[], 3),[],
2);
[dummy, d] =max (min (G (x , :  , : )+V{k+1}(F (x , : , : ) ),[],2),[],
3);
```

# Solving finite zero-sum games with MATLAB

- For reference, on a laptop with a Intel Pentium Mobile running at 1.6GHz with 1GB of RAM, the computation of the cost-to-go using backward iteration for a game with **one 100,000 states**, **10 actions** for each player, and **10 stages** takes about **40**seconds.
- When this procedure fails because Vminmax and Vmaxmin differ, one may want to use a mixed policy using a linear program. The indices of the states for which this is needed can be found using k=find(Vminmax-=Vmaxmin)